Network Utility Maximisation with Packet Corruption

Lachlan L. H. Andrew and David A. Hayes

This draft is a work-in-progress.

***Abstract*— Find out how much a source should back off its rate if some of its packets are corrupted.**

I. Introduction

TCP is often used over links with significant loss not caused by congestion, such as wireless links. To handle such links, TCP variants have been proposed which do not respond to loss in the way standard TCP does [1]. DCCP provides a mechanism to signal that packets have been corrupted. Even if corruption loss can be distinguished from congestion-related loss, loss causes inefficiency, and the question arises, what rate a flow should send at for a given level of loss. This paper finds the transmission rates which maximise the performance ("utility") of the network in the presence of loss, and remarkably show that flows with loss should often transmit at a higher rate than those without loss, while to achieve proportional fairness, corruption-related loss should be ignored entirely.

The paper is structured as follows: an overview of related work that addresses the problem of erasures in TCP type environments, followed by an extension of the NUM framework to cater for erasures which includes two case studies, retransmitting erased packets and Forward Error Correction (FEC), and finally a conclusion of the findings.

II. Related Work

TCP carries most flow controlled traffic across the Internet. The flow rate of modern TCP has been found to be proportional to $1/\sqrt{p}$, where p is the probability of packet loss [2], [3]. Since standard TCP uses packet loss as a signal indicating congestion, packet loss due to corruption prevents sources reaching their "ideal" rate. Wireless communication links have drawn attention to the question of what rate Internet traffic sources should send when the corruption probability is relatively high.

A. Loss-tolerant TCP

Over the last decade there have been many proposals on how to make TCP more tolerant to erasures that are not due to congestion. Some proposals, such as [4], isolate the error prone link from the rest of the network and correct link errors locally. TCP Westwood[5] operates in a similar fashion to TCP=RENO, except that when duplicate or a timeout occurs, `ssthresh` is not halved, but set to an estimate of the available bandwidth based the average rate ACKs were being received just prior to the loss. FAST-TCP [6] and TCP Vegas[7] seek to decouple the congestion and error control mechanisms using delay as an indicator of congestion. LT-TCP [1] uses Explicit Congestion Notification (ECN) signals and a hybrid FEC/ARQ to help tolerate erasures due to corrupted packets. TCP-ELSA [8] uses Explicit Loss Notification Messages (ELN) from a wireless base station to indicate a transmission based loss, as opposed to a congestion based loss, only invoking the TCP congestion control algorithm when the loss is due to congestion.

FEC was at one time considered computationally impractical for transport layer protocols implemented in software. Higher speed computers, and the use of erasure codes rather than full FEC codes have made them a practical tool.

III. Network Utility Maximisation framework

Let $U_i(x_i;\epsilon_i)$ be the total "utility" obtained by a user i which sends data at rate x_i [bits/s] ($x_{i\_\min} \le x_i \le x_{i\_\max}$), of which a fraction ϵ_i is corrupted by the network. In this analysis it will be assumed that ϵ_i is a parameter that is independent of optimisation process.

Kelly [9], [10], followed by Low [11], pioneered the approach of determining rates by maximising the sum of the utilities obtained by users, subject to the constraint that no link is overloaded. That is, solving

$$\max_x \sum_i U_i(x_i;\epsilon_i) \tag{1}$$

$$s.t. \quad Rx \le C \tag{2}$$

where $C = (c_l)$ with c_l the capacity of link l, and $R = (r_{li})$ with $r_{li} = 1$ if user i's data uses link l.

In the original work, ϵ was implicitly assumed to be 0, yielding $U_i(x_i)$ as a function of one variable. The problem is decomposed through the dual problem of maximising the Lagrangian

$$\mathcal{L}(x,p) = \sum_i U_i(x_i) - \sum_l p_l \left(\sum_i r_{li} x_i - c_l \right), \tag{3}$$

over $x \ge 0$ given p, and then minimising over $p \ge 0$. The Lagrange multipliers, p_l, can be thought of as a "price" that the network assigns to each link.

Each source solves its own sub-problem

$$\frac{\partial \mathcal{L}(x,p)}{\partial x} = U_i'(x_i) - \sum_l r_{li} p_l = 0 \qquad \forall i \tag{4}$$

by setting

$$x_i = (U_i')^{-1}(q_i) \tag{5}$$

where $q_i = \sum_l r_{li} p_l$ is the sum of the prices of the links it uses.

Similarly, each link solves its part of the dual problem, setting

$$\frac{\partial \mathcal{L}(x,p)}{\partial p_l} = \sum_i r_{li} x_i - c_l = 0 \qquad \forall l. \tag{6}$$

For example, this can be achieved by a gradient projection method, giving

$$p_l(t+1) = \left[p_l(t) - \gamma \left(\sum_i r_{li} x_i - c_l \right) \right]^+ . \tag{7}$$

where $[\cdot]^+ = \max(\cdot, 0)$ and $\gamma \in (0,1]$ is a step size parameter.

Note that users need not all have the same utility function, although they typically typically will to ensure fairness. This paper assumes all users have the same utility function.

IV. Utility maximisation with corruption loss

Assumptions

- Corrupt packets still cause congestion. This is true in the common case that the loss is on a last-hop wireless access link. It is not true in the other common case of a multi-hop mesh network.
- Sources can identify the fraction of loss due to corruption. Where stated, it will also be assumed that individual losses can be classified as congestion related or corruption related; in the remainder of the paper, it is only required that the fraction be known.
- Corruption probability is unrelated to congestion level. That is not the case if corruption is due to excess collisions, or the physical layer adapts its coding according to congestion.
- Links have fixed capacity. Again, this depends on the physical layer; on short timescales, it may not be true due to MUD or adaptive coding.
- Link prices are zero unless the link is fully utilised, in which case link prices are non-negative.
- Greedy sources,

If erasures occur, each user's utility is reduced by an amount depending on the erasure rate. The optimal rates now become

$$x_i = (U_i')^{-1}(q_i; \epsilon_i) \tag{8}$$

where $U_i'(\cdot;\epsilon) = \partial U_i(x_i;\epsilon)/\partial x_i$ and $(U_i')^{-1}(\cdot;\epsilon_i)$ is the inverse of U_i' with respect to the first variable.

The optimal rate to send in the presence of erasures depends only on how much a non-zero erasure rate reduces the user's utility. We now consider two cases.

This work asssumes that corrupt packets still cause their share of congestion, travelling through the network to the receiver. This is typical of download over wireless access links where corruption is most likely on the final link to the receiver, however, it may not necessarily follow where corruption occurs elsewhere on the path through the network.

A. Case 1: Retransmit corrupt packets

If no bit in a corrupt payload is trusted, the obvious response is to discard the packet and ask for it to be retransmitted. That is, packets are as good as lost, and we simply know that the "loss" is not due to buffer overflow. Thus, though a source transmits at rate x_i, it only achieves utility for rate $x_i(1-\epsilon_i)$:

$$U(x_i; \epsilon_i) = U(x_i(1-\epsilon_i); 0). \tag{9}$$

If erasures are carried through the network, x_i is still the rate from source i seen through the network. The Lagrangian is

$$\mathcal{M}(x,p) = \sum_i U_i(x_i(1-\epsilon_i);0) - \sum_l p_l \left(\sum_i r_{li} x_i - c_l \right) \tag{10}$$

giving the optimality conditions for the dual problems as

$$\frac{\partial \mathcal{M}(x,p)}{\partial x} = (1-\epsilon_i) U_i'(x_i(1-\epsilon_i);0) - \sum_l r_{li} p_l = 0 \quad \forall i \tag{11}$$

and

$$\frac{\partial \mathcal{M}(x,p)}{\partial p_l} = \sum_i r_{li} x_i - c_l = 0 \qquad \forall l \tag{12}$$

The price update rule to enforce (12) is identical to the one if there were no erasures (6).

Since $q_i = \sum_l r_{li} p_l$, (11) can be rewritten as

$$q_i = (1-\epsilon_i) U'(x_i(1-\epsilon_i); 0)$$

solving for x_i gives

$$x_i = \frac{1}{1-\epsilon_i} (U')^{-1} \left(\frac{q_i}{1-\epsilon_i}; 0 \right) .$$

Thus, by (8),

$$(U')^{-1}(q_i;\epsilon_i) = \frac{1}{1-\epsilon_i} (U')^{-1} \left(\frac{q_i}{1-\epsilon_i}; 0 \right) . \tag{13}$$

B. Case 2: Sources use "channel coding"

An alternative to asking for packets to be retransmitted is to add redundancy to the transmitted data in the form of forward error control coding (FEC) [**?**].

Consider again the case in which the corrupted packet carrys no useful information. Burst erasure correction codes, such as linear block codes [12], may be constructed to correct up to $n-k$ losses, where n is the number of encoded blocks, and k the number of encoded blocks required to retrieve the sent data. If packet erasures occur with probability, ϵ, and the erasure correction code perfectly matches network erasures $(n-k)/n = \epsilon$, the utility again becomes (**??**).

In practice we require $(n-k)/n > \epsilon$ to account for the random nature of packet erasures. This is less efficient than simply retransmitting corrupt packets, however, transmission across networks with high bandwidth-delay products benefit from the quicker correction of erasures. A robust mechanism will still need to be combined with an ARQ mechanism.

Consider again the case in which the corrupted packet carries no useful information. This forms an information-theoretic burst-erasure channel. If bursts are removed by interleaving data, each bit position becomes a binary-erasure channel with erasure probability ϵ. Since this binary-erasure channel has a capacity of $1-\epsilon$ bits per symbol, the utility again becomes (9).

In some applications, such as uncoded perceptual data like PCM audio, partically corrupt packets may still be useful. If corrupt packets on average carry useful information δ_i then

$$U(x_i; \epsilon_i, \delta_i) = U(x_i(1-(1-\delta_i)\epsilon_i); 0). \tag{14}$$

Except where noted, this is equivalent to a system which gets no value from corrupt packets, but has a slightly lower loss rate, $\tilde{\epsilon}_i = (1-\delta_i)\epsilon_i$.

From here on, we will consider a single source, and drop the subscript i.

C. Interpretation

The factor of $1/(1-\epsilon)$ can be interpreted as the source getting to retransmit the packets "for free". However, the congestion price must also be scaled up. In particular, for a lossed based protocol, each congestion-related loss must carry the weight of $1/(1-\epsilon)$ lost packets.

This is the only place in the paper in which (a) individual losses must be able to be classified as corruption/congestion (b) a system in which corrupt packets have equal value δ is different from a system with reduced corruption loss rate.

The above change can be implemented in a similar way to HS-TCP and H-TCP, by simply changing the factor by which the window is reduced in response to each loss, while keeping the additive increase rate unchanged. The difference would be that the factor depends on ϵ rather than the window size or time since the last loss. However, increasing the factor by which the window is reduced increases the burstiness of the traffic. Simpler solutions can be obtained for special cases of the utility function, U, as described in Section VI.

V. Conditions for loss to reduce fair rate

The intuitive behaviour of corruption errors always reducing the fair rate corresponds to

$$\frac{1}{1-\epsilon}(U')^{-1}\left(\frac{q}{1-\epsilon};0\right) \le (U')^{-1}(q;0). \tag{15}$$

for all $q>0$ and $\epsilon \in [0,1)$. Call a utility function satisfying (15) *rate reducing*. A utility function satisfying (15) with the inequality reversed is *rate increasing*. If it is both rate increasing and rate decreasing, it is *rate neutral*. It is possible for a utility function to be neither rate increasing, rate decreasing nor rate neutral.

Theorem 1: Let $g(r) = (U')^{-1}(1/r)$.

A utility funciton is rate reducing if and only if

$$ag(r) \ge g(ar) \tag{16}$$

for all $a \in (0,1]$ and $r>0$. A sufficient but not necessary condition for U to be rate reducing is that g be convex on $[0,\infty)$.

A utility funciton is rate increasing if and only if

$$ag(r) \le g(ar) \tag{17}$$

for all $a \in (0,1]$ and $r>0$. A sufficient but not necessary condition for U to be rate increasing is that g be concave on $[0,\infty)$.

Proof: Setting $a = 1-\epsilon$ and $r = 1/q$ in (15) yields (16). To see that convexity of g is sufficient, note that $g(0)=0$ since.... To see that it is not necessary, consider

$$g_1(r) = \begin{cases} r^2 & 0 \le r \le 1 \\ 3r/2 - 1/2 & r > 1. \end{cases}$$

The conditions for rate increasing follow similarly, using $g_2(r) = g_1^{-1}(r)$ to show that concavity is not necessary. ■

VI. Mo and Walrand's α-fairness

A useful family of utility functions introduced in [13] has the form

$$U(x) = \begin{cases} \log x & \alpha = 1 \\ (1-\alpha)^{-1}x^{1-\alpha} & \alpha > 1 \end{cases} \tag{18}$$

giving

$$U'(x) = x^{-\alpha} \tag{19}$$

For $\alpha = 1$, this yields Kelly's proportional fairness [10], for $\alpha \to \infty$ it yields max-min fairness and for $\alpha \to 0$ it seeks to maximise throughput without regard to fairness. Most importantly, in the "minimum potential delay" case [14] corresponding to $\alpha = 2$, it yields a good approximation to TCP Reno's behaviour [15]. Similarly HS-TCP [16] targets $\alpha = 1.2$.

For these utility functions, (13) and (19) give

$$\frac{1}{1-\epsilon}(U')^{-1}\left(\frac{q}{1-\epsilon};0\right) = (1-\epsilon)^{1/\alpha-1}q^{-1/\alpha} \tag{20}$$

$$= (1-\epsilon)^{1/\alpha-1}(U')^{-1}(q;0) \tag{21}$$

The optimal rate is simply obtained by calculating the congestion window, w, based only on the packets received and ignoring corruption loss, but then instead of using w directly in the sliding window mechanism, using

$$\hat{w} = (1-\epsilon)^{1/\alpha-1}w. \tag{22}$$

For example, TCP would increase CWND by 1 every RTT as usual, and halve CWND on loss as usual, but scale CWND by $(1-\epsilon)^{1/\alpha-1}$ to obtain $\hat{w}$ which is then the actual number of packets are allowed to be outstanding in the network. This incurs minimal additional computation, requires no table lookup, and does not increase the burstiness of the rate.

A. Interpretation

Corollary 1: A function of the form (18) is rate increasing if and only if $\alpha \ge 1$, rate decreasing if and only if $\alpha \le 1$ and rate neutral if and only if $\alpha = 1$.

Several special cases are of interest.

a) Max-min fairness, $\alpha \to \infty$: Since Max-min fairness aims to give a high rate to the flow with the lowest rate, regardless of the cost incurred, we would expect $\alpha \to \infty$ to give a fair rate, x, which depends only on the throughput achieved, $x(1-\epsilon)$, without regard to the impact of the loss ϵ on the price q. Indeed, (22) becomes $\hat{w}_\infty = w/(1-\epsilon)$. In the terminology of Section IV-C, this allows sources to retransmit lost packets "for free", without the concomitant requirement that they respond more severly to losses.

b) Maximum throughput, $\alpha \to 0$: Corruption losses waste capacity, and intuitively maximum throughput should allocate all bandwidth to flows on a given link with the lowest loss rates.

In the case that some flows at each link have zero loss, $(1-\epsilon)^{1/\alpha-1} = 1$ for loss-free flows ($\epsilon = 0$), while it tends to 0 for all other flows, and the bandwidth is allocated among the loss free flows as if the lossy flows did not exist.

The case is slightly more complicated if all flows at some bottleneck link experience some loss. In this case, $(1-\epsilon)^{1/\alpha-1} \to 0$ for all flows, and it seems that (22) predicts zero windows (and hence zero throughput) for all flows, which clearly does not maximise throughput. However, the network's response does indeed ensure maximum throughput by the way the congestion signal adapts. If the link were not fully utilised, the congestion signal (either delay or congestion loss) would drop to zero, and the calculated window w would rise until the link is fully utilised. For any non-zero α, there is are equilibrium rates that fill the link capacity. Because $(1-\epsilon_2)^{1/\alpha-1}/(1-\epsilon_1)^{1/\alpha-1} \to 0$ as $\alpha \to 0$ for $\epsilon_2 > \epsilon_1$, in the limit all bandwidth will be allocated to the flows with the lowest corruption loss, and throughput will indeed be maximised.

c) Proportional fairness, $\alpha = 1$: For $\alpha = 1$ the optimal rate is completely independent of the erasure rate, ϵ.

FAST TCP [6] is a well-known protocol which uses delay rather than loss as a signal of congestion, and achieves proportional fairness [17], [18]. Early implementations of FAST responded to loss in the same way that Reno does, but more recent implementation totally ignore small amounts of loss. The results of this section show that that is actually the most consistent response, given its proportionally fair bandwidth allocation.

d) TCP fairness, $\alpha = 2$: Since "TCP friendliness" corresponds to $\alpha = 2 > 1$, TCP is on the max-min fair side of proportional fairness, and again it is in keeping with that flows should send at rate $(1-\epsilon)^{-1/2}$ times *higher* when they experience non-congestion packet corruption on their final link.

This finding goes against the traditional notion that corruption loss should cause TCP-friendly flows to reduce their rates. That notion comes from assuming that the aim of TCP-friendliness is to maximise throughout, whereas it is actually to promote fairness between the rates flows achieve.

References

[1] O. Tickoo, V. Subramanian, S. Kalyanaraman, and K. K. Ramakrishnan, "LT-TCP: End-to-end framework to improve TCP performance over networks with lossy channels." in *IWQoS*, 2005, pp. 81–93.

[2] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. Networking*, vol. 5, no. 3, June 1997.

[3] M. Mathis, J. Semke, and J. Mahdavi, "The macroscopic behaviour of the TCP congestion avoidance algorithm," *ACM Computer Communication Review*, vol. 27, no. 3, 1997.

[4] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, "Performance enhancing proxies intended to mitigate link-related degradations," IETF, RFC 3135, June 2001.

[5] S. Mascolo, C. Casseti, M. Gerla, M. Sanadidi, and R. Wang, "TCP Westwood: end-to-end bandwidth estimation for efficient transport over wired and wireless networks," in *Proc. ACM Mobicom*, Rome, Italy, July 2001.

[6] D. X. Wei, C. Jin, and S. H. Low, "FAST TCP: Motivation, architecture, algorithms, performance," *IEEE/ACM Trans. Networking*, vol. 14, no. 6, pp. 1246–1259, Dec. 2006.

[7] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End-to-end congestion avoidance on a global internet," *IEEE J. Select. Areas Commun.*, vol. 13, no. 8, pp. 1465–1480, Oct. 1995.

[8] Y. Yang, H. Zhang, and R. Kravets, "Channel quality based adaptation of TCP with loss discrimination," in *Proc. of the IEEE Global Communications Conference (GLOBECOM'02)*, 2002. [Online]. Available: citeseer.ist.psu.edu/yang02channel.html

[9] F. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, 1997.

[10] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *J. Op. Res. Soc.*, vol. 49, pp. 237–378, 1998.

[11] S. H. Low and D. E. Lapsley, "Optimization flow control I: Basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, no. 6, pp. 861–875, Dec. 1999.

[12] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM Comp. Commun. Rev.*, vol. 27, no. 2, pp. 24–36, Apr. 1997.

[13] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Networking*, vol. 8, no. 5, pp. 556–567, Oct. 2000.

[14] L. Massoulie and J. Roberts, "Bandwidth sharing: objectives and algorithms," *IEEE/ACM Trans. Networking*, vol. 10, no. 3, pp. 320–328, June 2002.

[15] S. Kunniyur and R. Srikant, "End-to-end congestion control: utility functions, random losses and ECN marks," *IEEE/ACM Trans. Networking*, vol. 11, no. 5, pp. 689–702, Oct. 2003.

[16] S. Floyd, "Highspeed TCP for large congestion windows," IETF, RFC 3649, Dec. 2003.

[17] S. H. Low, L. L. Peterson, and L. Wang, "Understanding Vegas: A duality model," *Journal of the ACM*, vol. 49, no. 2, pp. 207–235, Mar. 2002.

[18] L. L. H. Andrew, L. Tan, T. Cui, and M. Zukerman, "Fairness comparison of FAST TCP and TCP Vegas," in *Proc. Intl. Teletraffic Congress 19 (ITC-19)*, 2005.